Alternative Approaches for Solving the Sensor Placement Problem in Large Networks

R. Pinzinger*, J. Deuerlein**, A. Wolters** and A. R. Simpson****

*3S Consult GmbH, 76133 Karlsruhe, Mathystraße 14-16, PH +49 721 335 0337, pinzinger@3sconsult.de
**3S Consult GmbH, 76133 Karlsruhe, Mathystraße 14-16, PH +49 721 335 0336, j.deuerlein@3sconsult.de
***3S Consult GmbH, 80333 Munich, Karlstraße 19, PH +49 89 5404146 50, wolters@3sconsult.de
****School of Civil, Environmental and Mining Engineering, University of Adelaide, Adelaide SA 5005, PH +61 8 8303 5874, angus.simpson@adelaide.edu.au

ABSTRACT

Positioning sensors in a water supply network is a NP–hard task. We propose three algorithms – one based on integer linear programming (ILP) and the other two based on the Greedy paradigm. We apply these algorithms to real case networks and compare the results of these algorithms with the results of an algorithm based on NSGA II, a genetic algorithm. We come to the conclusion that our algorithms outperform NSGA II in every single case. The algorithm based on linear integer programming may be applied as a competitor to the algorithm implemented in TEVA –SPOT (Berry, 2009), while the first Greedy algorithm may replace the ILP algorithm in large networks due to its faster running time. The second Greedy algorithm approaches the question on finding those nodes which are the most sensitive to variations in pressure and are thereby ideal places to monitor the hydraulic state of a water distribution network.

KEYWORDS

Graph Theory, Sensor location layout, Greedy Algorithm, Genetic Algorithm, Integer Linear Programming, Sensitivity

INTRODUCTION

Modern water supply systems consist of pipes, pumps, storage tanks and valves. Since these networks span over wide areas they often require large investments and thus it is desirable to operate these systems with as little loss as possible and save energy and material costs to the greatest possible extent. Furthermore, especially in the century after the terroristic event of 9/11, consumer security is also an important issue. Under these circumstances, it is natural to extend the water supply infrastructure by installing sensors that monitor the hydraulics of the network as well as the quality of the supplied drinking water.

The benefits from monitoring the hydraulics lie in getting to know, where the water moves in the network and what is the pressure. From these data, one could deduce rehabilitation plans, strategies of how to operate the pumps in the system or how to

set the valves. In addition one can detect pipe bursts by simply checking how the pressure in the network evolves. If water distribution operators monitor water quality parameters such as chlorine, pH–level, contamination by natural causes or by terrorists, they can decide the best way to warn the public and thereby prevent adverse impacts including disease or even deaths.

The German Ministry of Education and Research is funding a project called IWaNet, which is short for Intelligent Water Network. It includes the development of a hybrid network analyzer that consists of a deterministic hydraulic real-time simulation model (DSM) combined with an artificial neural network (ANN). Both are fed by real time data coming from multi-parameter sensors that monitor the network hydraulics as well as water quality at selected locations in the system. Whereas actual demands, tank water levels and pump operations are used as boundary conditions of the hydraulic solver the ANN estimates the current state of the system based on the measurements. Since the ANN can handle only a limited number of data points that are actually the monitoring stations, the connection with the DSM fills this gap. Against that background it is evident that the locations of the monitoring points have to be chosen very carefully in order to get the most information possible from the real-time data.

From a theoretical point of view the question as to where to install sensors is very hard and many scientists have been working in this field: including Berry et al. (2010) who proposed solving an Integer Linear Programming problem (ILP). This type of programming definition is very hard to solve and computationally expensive. However, Integer Linear Programming has a significant advantage in that if it terminates an optimal solution is reached. Additionally, one can prove the optimality of the solution. Grayman et al. (2006) applied a Genetic Algorithm (GA) to this task. Genetic algorithms are types of random search algorithm that mimics evolution by using individuals whose genome encodes information which is passed according to mating rules to the next generation. Although they are very popular for real world data sets they sometimes work very slowly. Krause et al. (2008) propose a Greedy algorithm. Greedy algorithms are popular since they are always simple to implement, run fast and produce solutions which most times are close to the real optimum.

In the next part of the paper, we want to introduce two approaches which both adapt the greedy paradigm and an Integer Linear Programming approach. We will describe how these algorithms have been constructed and compare them to NSGA II (Deb, 2001), an evolutionary algorithm, on 4 different sized water distribution systems.

ILP–APPROACH FOR MAXIMUM COVERAGE

Recently, the TEVA – SPOT (Threat Ensemble Vulnerability Assessment – Sensor Placement Optimization Toolkit) has been published by the US EPA. It provides algorithms for finding ideal sensor locations. The objective is to minimize the impact of randomly chosen intrusion scenarios. A two step scheme has been chosen: in the first step, a Markov chain Monte Carlo simulation is run, during which several intrusion scenarios are simulated concerning how dangerous substances spread in a water distribution system. These scenarios are each weighted with each individual sce-

nario's probability of occurrence. In the second step, one solves an ILP which minimizes the overall impact by finding the ideal placement of the sensors considering the simulated attack scenarios. The problem with this approach is, that for the first step a great number of hydraulic simulations have to be run and kept in storage. This accumulates even for small networks to very large sizes of data and days of computation time.

In our approach the objective is to find a sensor configuration that maximizes the detection likelihood of intrusions at any location in the network. At the beginning for every node *i* in the network the set of nodes *U(i)* upstream of *i* is calculated using a Breadth–First–Search (BFS). A node *j* is considered to be upstream of a node *i* if and only if the travel time from *j* to *i* is within a given time span $t_{\max}$; in that case $j \in U(i)$. As a simplification we assume perfect mixing at junctions and perfect sensors. With the set *U(i)* we know the locations from where water quality information can travel to *i* within $t_{\max}$.

The computations are based on the results of an extended period simulation of the network. In our case, we simulated the hydraulics of a day using 15 minute time intervals. As mentioned above the BFS only places nodes in the queue if their temporal distance to *i* is smaller than $t_{\max}$. For the BFS, the distance from every node to *m* is initially set to a large number, say one billion seconds, whereas the distance from *i* to itself is obviously 0 seconds. During the BFS, the time the water takes to pass from one endpoint of a pipe to the other endpoint is computed. For that purpose we use the results of the extended period simulation considering that the flow velocity of the water varies with time. Based on this preliminary calculation we can apply a version of the Floyd–Warshall–algorithm (Floyd, 1962) in order to compute the minimal flow time of the water from an arbitrary node *l* to a sensor *i.* That computation is repeated for every single time interval of the simulation horizon.

It follows that for every node *l* that is upstream of *i* the maximum time the water takes to flow from *l* to *i* over all time intervals is considered. If that maximum is smaller than the maximum time span $t_{\max}$ then node *l* is observed by node *i*. Let *n* be the number of nodes and *V* be a binary $(n \times n)$-matrix. The element (*i,l*) of *V* is set to 1 if node *l* is observed by node *i* otherwise it is set to 0. By doing this we trim the area observed by a quality sensor at node *i* to a minimal set of nodes that is monitored around the clock. Thus, we ignore those nodes, which are observable only for some time span during the cycle of a day.

The matrix *V* is used for the formulation of the following ILP. Let *k* be the maximum number of sensors, let *x* be a binary vector of dimension *n* where x_i is 1 if a sensor is placed at node *i* and 0 otherwise. Let *b* be a binary vector of dimension *n* where b_i is 1 if node *i* is observable by the sensor configuration in *x* and 0 otherwise. Let *c* be a vector of dimension *n* where every entry is 1. Then we can formulate the task of finding best locations for sensors as an Integer–Programming–Problem as follows:

$$\max_{b,x} c^T b \quad (1)$$

subject to

$$\frac{1}{n} V^T x - b \leq 0 \quad (2)$$

$$-V^T x + b \leq 0 \quad (3)$$

$$c^T x \leq k \quad (4)$$

$$x, b \in \{0,1\}^n \quad (5)$$

x is the binary vector of possible sensor locations, b is the binary vector of observed nodes.

The product $\frac{1}{n} V^T x$ results in a vector, whose entries are all between 0 and 1, since $V^T x$ counts for every node i how many sensors of configuration x observe i. This count is normalized by multiplying by $\frac{1}{n}$ because no node can be monitored by more than the number of nodes in the network. Furthermore, one entry i of $\lambda V^T x$ is greater than 0 if and only if the current configuration of x provides at least one sensor which observes node i. Therefore, the first constraint (Eq. 2) ensures that if the configuration of sensors x observes node i the i-th entry of vector b is 1. The second constraint (Eq. 3) guarantees that if a configuration of sensors x does not observe node i the i–th entry of b is 0.

Using these two constraints we force b to include all nodes that are observed by configuration x. Additionally, we prohibit b from including anything that is not observed by x. Constraint three (Eq. 4) assures that we do not employ more sensors than we agreed upon before. To solve this problem usually algorithms like branch and bound or Gomory Chvatal cutting planes are used. In this case the package LPsolve (Berkelaar 2010) which uses branch and bound approach has been chosen.

In contrast to TEVA – SPOT we compute for every time interval during a day a BFS for every node in the network in order to find the nodes upstream of a given node within the aforementioned time span $t_{\max}$. The results of these computations are distilled down to matrix V. As a consequence we do not have to handle vast amounts of data. Furthermore, the computational complexity of a BFS run is $O(n^2)$ in a graph with n nodes. The computational complexity of the computation of matrix V is therefore $O(dn^3)$ since we compute the BFS for every node and d time intervals of a day.

In TEVA – SPOT s arbitrarily chosen extended period water quality calculations have to be carried out for. In contrast, the approach presented above is based on one single extended period simulation resulting in much shorter calculation time. However, the reaction kinetics are not considered.

GREEDY ALGORITHM FOR MAXIMUM COVERAGE

Since solving large ILP problems is very time consuming, we formulated a Greedy algorithm which computes an approximation to the ideal cover of measurement locations. For this Greedy algorithm, we use the same precomputation of matrix V as in the ILP–approach above. Then, we make the Greedy assumption that we find the best coverage of nodes in the network using k nodes by iteratively following a simple scheme: find that row r of V that has the most 1 entries. That row corresponds to the node r that monitors the most nodes. Mark all those nodes, which are monitored by node r and neglect them in the following computations. Find in the resulting matrix V' that row r' with the most 1 entries, mark the corresponding nodes, so that they are ignored in future computations and iterate. Finding the row with the most 1 entries is an $O(n^2)$ task, and therefore the Greedy algorithm is of $O(dn^3)+O(kn^2)=O(dn^3)$ complexity.
As will be seen in later parts of this paper, the Greedy algorithm for maximum coverage works very fast and produces, at least in the four test cases we considered, solutions which are only slightly worse than the ideal answer of the aforementioned ILP–approach. Since it is faster than the ILP–approach and only slightly worse, we propose to use this algorithm for bigger networks (10,000 nodes or more).

GREEDY ALGORITHM FOR FINDING PRESSURE SENSITIVE LOCATIONS

Observation of pressure in a water distribution network can support the detection of line breaks in real time. Using a calibrated numerical model, the results of hydraulic steady-state calculations can be compared to online measurement data. If at some locations the pressure in the physical system suddenly drops below the results of the computation, this can be an indication for the existence of a new leak in the system.
The comparison of measured data and data predicted by the hydraulic model can be done more efficiently by an artificial neural network (ANN) that has been trained in advance by numerous simulation results of a number of leakage scenarios. Based on these training data sets the artificial neural network is able to predict the location and size of the leakage. For this, we need measurements in highly sensitive places. The more the measured value of a chosen location reacts to changes in the model parameter, the more valuable are the measurements.

In the following the sensitivity of nodal pressure as to changes in demands or other outflows like leakage is considered as an example.

The hydraulic steady-state can be formulated as a nonlinear optimization problem (e.g. Todini 1987). Using sensitivity results of nonlinear programming the sensitivity of the nodal heads H as to a change in withdrawals at the nodes Q can be computed by

$$\nabla_Q H = \left(A^T \frac{1}{\alpha} D^{-1} A \right)^{-1}$$

where A is the (m x n)- incidence matrix of the graph (n: number of nodes, m: number of pipes), α is the exponent of the hydraulic headloss equation and D is a diago-

nal matrix, whose entries are the derivatives of the hydraulic head-loss equations of the pipes.

$\nabla_Q H$ is the $(n \times n)$ sensitivity matrix. The entry $\nabla_Q H_{ij}$ at position i, j represents how sensitive the pressure at node *i* is to a change in outflow at node *j*.

The sensitivity of the pressure at node *i* against outflow changes at any other node in the network can be estimated by $\sum_{j=1}^{n} \nabla_Q H_{i,j}$. The node that maximizes that sum is the most sensitive. So far we have been considering just one node. Usually for a measurement program a limited number of sensors are available. The calculation of the most sensitive node can be generalized by the following iterative approach:
Firstly, we compute the sensitivity matrix for the entire system and search for the row *i* that has the biggest row sum value. In the next step "infinity" is added to the *i–th* diagonal element of the matrix $A^T D^{-1} A$. This sets the sensitivity of node *i* to zero for the next inversion of the matrix. With this modified sensitivity matrix the second most sensitive node is determined. The procedure is repeated until the maximum number of sensors available is reached. Doing this, we iteratively set the sensitivity of the most sensitive nodes and their neighbors to zero and make sure, that the result of our computations is a set of nodes which are spread all over the network.
Since we have to invert a matrix for every step of our search, this algorithm has $O(kn^3)$ complexity in every step where *n* is the number of nodes and *k* is the number of sensors.

We developed this program to find good locations for multi-parameter sensors. The ILP–approach and the Greedy algorithm for maximum coverage compute sensor locations which monitor the quality of the water in wide areas of the network, whereas this program computes the best locations in order to monitor the hydraulics of the network.

COMPARISON OF OUR ALGORITHMS WITH NSGA II

In this section a comparison is made of the ILP–approach and the Greedy algorithm for maximum coverage with NSGA II based on consideration of their running time for the coverage problem. We applied each of the algorithms to four real case networks. The size of the networks ranges from about 300 nodes up to 5,000 nodes.

Table 1: Comparison of computation time for the ILP - approach, the proposed Greedy–algorithm for maximum coverage and NSGA II

	ILP	Greedy	NSGA II
Town_300	1s	1s	1560s
Town_1000	4s	1s	3760s
Town_2000	17s	1s	10640s
Town_5000	111s	5s	60210s

Since both, the Greedy algorithm for maximum coverage and the ILP–approach only produce a single solution they cannot be compared directly to NSGA II. NSGA II determines a so called Pareto front which is a set of solutions which do not dominate each other. Thus we will only compare the results from the Greedy algorithm and the ILP–approach directly in the following table. We want to compare the number of nodes that are covered by the computed sensor network.

Table 2: Comparison of the ILP-approach and the Greedy-algorithm for maximum coverage concerning how many nodes a computed sensor network layout can monitor

	Number of nodes	Number of sensors	ILP number of covered nodes	ILP percentage of coverage	Greedy number of covered nodes	Greedy percentage of coverage
Town_300	339	5	160	47,2%	158	46,6%
Town_1000	1261	8	679	53,8%	679	53,8%
Town_2000	2146	12	1293	60,3%	1286	59,9%
Town_5000	4923	21	2011	40,8%	2002	40,7%

From both tables 1 and 2, we can deduce that the Greedy algorithm works faster than the ILP–approach. Recall that the results from the ILP–approach are the optimal solutions. Thus the Greedy algorithm produces solutions that are very close to the optimal solutions.

We can conclude that for networks of less than 5,000 nodes that there is no need to use the Greedy algorithm since the computational effort for computing the solution using the ILP–approach is not much more than for the Greedy algorithm. Since the result of the ILP–approach is always optimal it is superior to the Greedy algorithm which will only approximate the real optimal solution. The proposed Greedy algorithm for maximum coverage may be used only for large networks.

Using NSGA II (Deb, 2001), we considered two conflicting objective functions: the first one produces its minimal value when the genome contains exactly the number of sensors we want to install, whereas the second objective function simply counts the number of nodes which are not covered by the sensor network which is coded in the genome. Both functions are to be minimized. Thus a sensor network configuration is sought that covers as many nodes as possible for the maximum number of sensors selected.

The following table contains the parameters used for the NSGA II computations:

Table 3: Settings for NSGA II

	Number of individuals	Number of generations	Crossover probability	Mutation probability
Town_300	100	100	0.5	0.0025
Town_1000	150	150	0.5	0.0007
Town_2000	250	150	0.5	0.00035
Town_5000	350	350	0.5	0.00015

In the following we compare the results of the Greedy algorithm for maximum coverage, the ILP-approach and the Pareto Front produced by NSGA II:

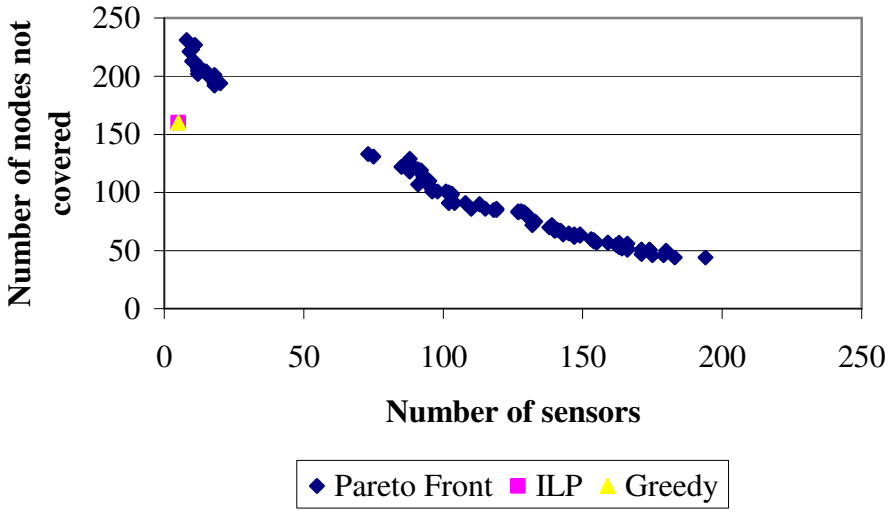

Figure 1: Comparison of results of Greedy-, ILP- and NSGA II-approach considering maximum coverage; Town_300

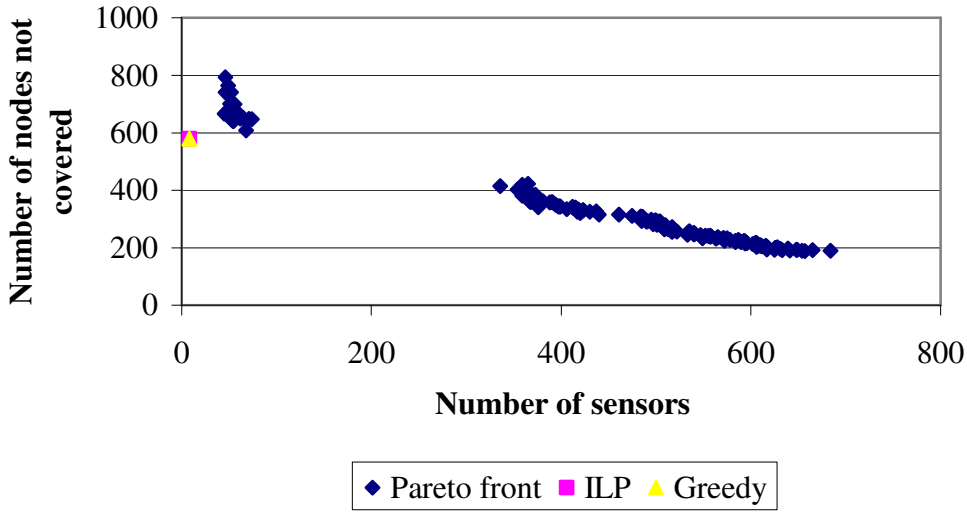

Figure 2: Comparison of results of Greedy-, ILP- and NSGA II-approach considering maximum coverage; Town_1000

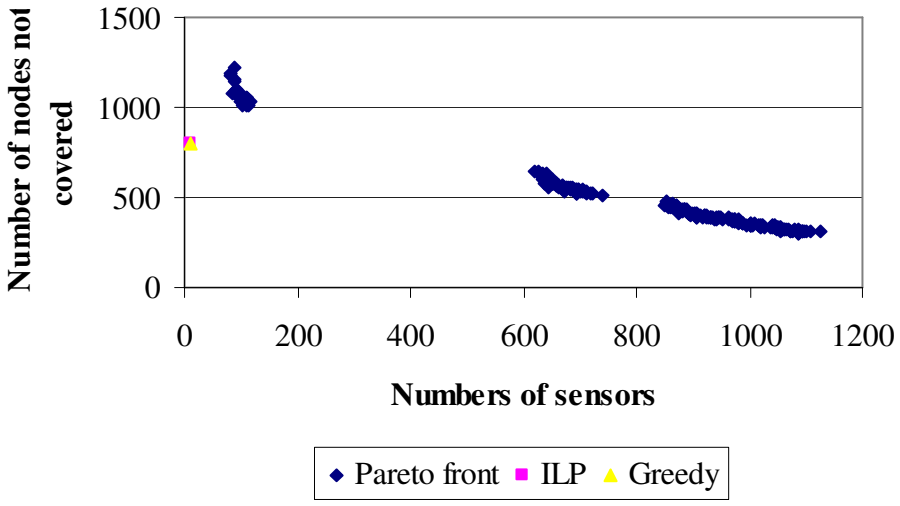

Figure 3: Comparison of results of Greedy-, ILP- and NSGA II-approach considering maximum coverage; Town_2000

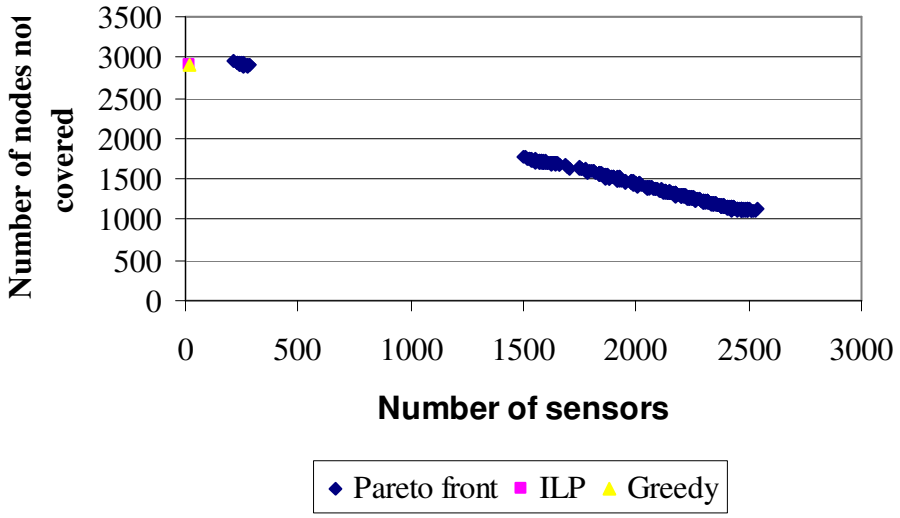

Figure 4: Comparison of results of Greedy-, ILP- and NSGA II-approach considering maximum coverage; Town_5000

In the Fig. 1 to 4 above, we can compare the quality of the computed solutions of the Greedy algorithm and the ILP–approach to the output of NSGA II considering maximum coverage. We see in all figures that NSGA II did not produce any solution that is able to monitor as many nodes as for the the Greedy algorithm or the ILP–approach for a small number of sensor nodes.

One might propose extending the number of individuals or number of generations (or both) so that NSGA II has an increased chance of finding improved solutions, but this will result in increased computation time.

CONCLUSIONS

In this paper, we have presented three algorithms for computing an optimal sensor network layout. These included an Integer Linear Programming approach and two Greedy algorithms. We compared these algorithms considering time for computation and the quality of the solution. We conclude, that each proposed algorithm produces good solutions within reasonable time.

For the comparison of the Greedy algorithm and ILP we chose a very specific objective function for both algorithms: that is to maximize that part of the network that can be observed by a fixed number of sensors. In future work, one has to develop objective functions with several aims: one aim might be to maximize security for civil and state – owned property, another aim might be, that every node that is visible by one sensor is at least visible by another one, so that every piece of information can be triangulated in the network.

ACKNOWLEDGEMENTS

The research presented in this paper is part of the project IWaNet (Support Code: 13N10623) that is funded by the German Ministry for Education and Research (BMBF).

REFERENCES

Berkelaar, M. (2010): lp_solve 5.5, attained from http://lpsolve.sourceforge.net/5.5/, visited on December 2010-12-03

Berry, J. (2005): Sensor Placement in Municipal Water Networks in Journal of Water Resources, Planning and Management, Volume 131, Issue 3, May/June 2005

Berry, J. (2009): TEVA-SPOT Toolkit Version 2.3, available at https://software.sandia.gov/trac/spot, visited on 2010-12-21

Deb, K. (2001): A fast and elitist multi – objective genetic Algorithm – NSGA – II in KanGAL Report Number 2000001, source Code attained from http://www.iitk.ac.in/kangal/codes.shtml, visited on 2010-12-03

Floyd, R. (1962): Algorithm 97, Communications the ACM Volume5, Number 6

Grayman, W. (2006) : Locating Monitors in Water Distribution Systems: Red Team – Blue Team Exercise in Journal of Water Resources, Planning and Management, Volume 132, Issue 4, July/August 2006

Krause, A. (2008): Efficient Placement Optimization for Securing Large Water Distribution Networks in Journal of Water Resources, Planning and Management, Volume 134, Issue 6, November/December 2008

Todini, E. & Pilati, S. (1987.): A gradient method for the analysis of pipe networks. International Conference on Computer Applications for Water Supply and Distribution, Leicester Polytechnic, UK, September 8-10.